Conversion Program: HTML to Natural

Not applicable to mainframe systems.

This section describes the use of HTML to Natural, a program that enables you to convert an HTML page into a Natural subprogram for use with Natural Web Interface.

Using HTML to Natural to generate Natural code from an HTML page avoids you having to adapt HTML input to the conventions of Natural code. You can then move the "HTML-page-turned-subprogram" to the server, including all the other Natural program logic you have added. If you want to change the HTML page again, go back to your source, convert it and move it to the server again. This is much easier than writing HTML with a browser, moving it to the server, adding Natural program logic and reiterating the process if your HTML page changes.

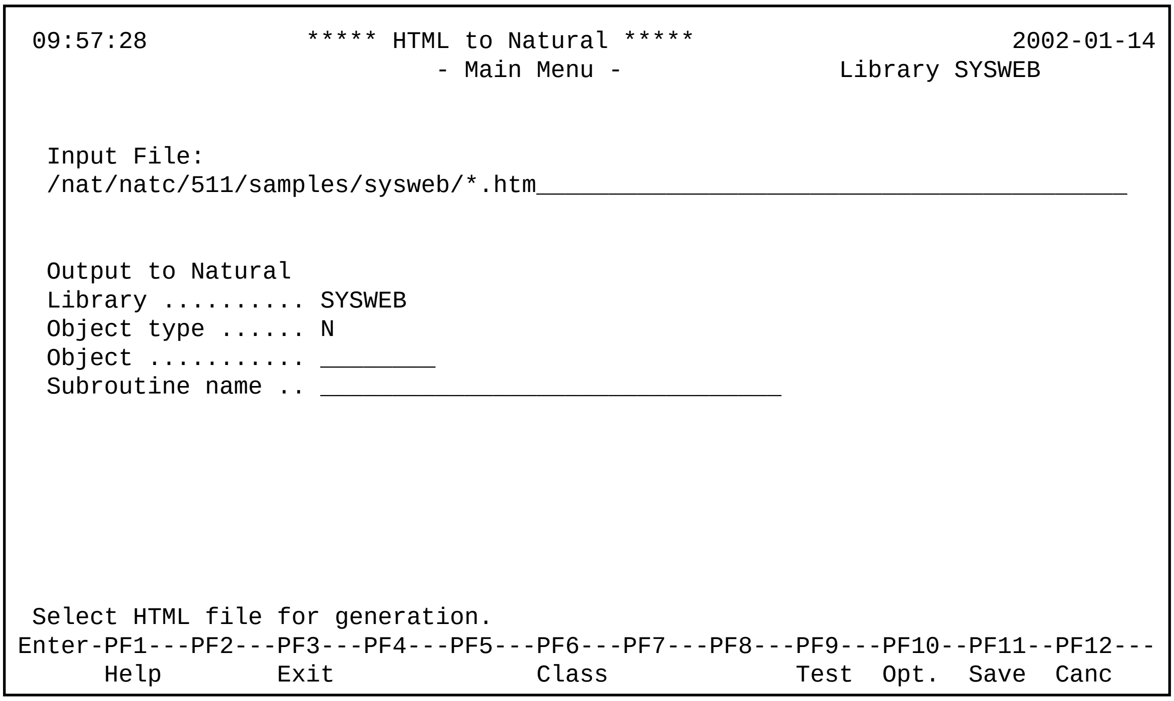
This section covers the following topics:

- Using the Conversion Program
- Inserting a Natural Tag
- Options
- Generating a DCOM Class
- Online Test Utility WEB-ONL

Using the Conversion Program

If your basic Web pages are designed with editing tools, it takes some effort to include such a page in a Natural subprogram that can be called from the Web.

"HTML to Natural" is a program that uses an HTML page as input and generates a Natural subprogram, which can be called by the Natural Web Server Extensions using the Natural Web Interface.



```
 09:57:28           ***** HTML to Natural *****                  2002-01-14
                             - Main Menu -              Library SYSWEB


  Input File:
  /nat/natc/511/samples/sysweb/*.htm_________________________________________


  Output to Natural
  Library .......... SYSWEB
  Object type ...... N
  Object ........... ________
  Subroutine name .. ________________________________








 Select HTML file for generation.
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit              Class             Test  Opt.  Save  Canc
```

Below is information on:

- Functions and PF Keys
- Generating a Subprogram/Subroutine to be called direct from the Web

Functions and PF Keys

PF Key	Function	Explanation
PF1	Help	Invokes the Help function for the field at which the cursor is positioned.
PF3	Exit	Leaves the program and returns to the command line.
PF6	Class	Starts the program that generates a DCOM Class (see the relevant section).
PF9	Test	Starts the Online Test Utility (see the relevant section).
PF10	Opt.	Options. Specifies options for the generation process.
PF11	Save	Saves the selected input and output files as default Natural parameters.
	Next	Starts generating the program.
ENTER		

Generating a Subprogram/Subroutine to be called direct from the Web

To generate a subprogram/subroutine to be called direct from the Web

1. Select your HTML page.
2. Close the Natural library you want to generate.
3. Select the object type you want to generate.
4. Select your Natural file name.
5. Start the generation.
6. After generation, you can call the Natural Web Interface to show the page.

Inserting a Natural Tag

If you use Natural on your HTML page, it is possible to specify your special Natural coding direct in the HTML page. After generation, the program needs no additional changes.

The HTML2NAT program can recognize a <NATURAL> tag. All lines between <NATURAL> and </NATURAL> will be copied, as they are, to the generated Natural source object.

Appearance

```
<NATURAL> </NATURAL>
```

Below is information on:

- Attributes DATA, LDA, GDA, SUB, NOT
- Comment Tag
- ASP-like Script Commands
- Additional Script Directives
- Example 1 of a Simple Generation
- Example 2 of a Simple Generation with a Natural Tag

Attributes DATA, LDA, GDA, SUB, NOT

Listed below are attributes provided to define coding sections that are to be moved within the program or excluded from the program.

Attribute	Explanation
DATA	<NATURAL DATA> or <NATURAL LDA> moves the defined section to the DEFINE DATA LOCAL part of your program.
LDA	
GDA	<NATURAL GDA> moves the defined section to the DEFINE DATA GLOBAL part of your program.
SUB	<NATURAL SUB> moves the defined section to the end of the program. This enables you to specify inline subroutines.
NOT	<NATURAL NOT> excludes the defined section from the program. This enables you to specify the design of part of a page that will be generated by a program.

Comment Tag

Use the comment tag "<!-- -->" to hide the display of defined sections of your coding. If you use the comment tag and <NATURAL NOT>, you can display the predefined page with a normal browser. This helps you to specify your page and replace parts of the page dynamically.

ASP-like Script Commands

With the new Natural Version 5.1 of HTML2NAT, not only <NATURAL> and </NATURAL> can be used but also ASP-like (Active Server Page) script commands which are differentiated from the text by using the "<%" and "%>" delimiters.

Additional Script Directives

The following Natural-specific directives must be used when writing a Natural subprogram:

Output directive: <%= ... %>
    Short form for <% PERFORM W3HTML ... %> tag
Subprogram directive: <%SUB ... %>
    equal to the <NATURAL SUB> ... </NATURAL> tag
Global Data Area directive: <%GDA ... %>
    equal to the <NATURAL GDA> ... </NATURAL> tag
directive: <%LDA ... %>
    equal to the <NATURAL LDA> ... </NATURAL> tag
Not directive: <%NOT ... %>
    equal to the <NATURAL NOT> ... </NATURAL> tag
Processing directive <%@ LANGUAGE=NATURAL %>
    indicates that the used language is Natural..

Example 1 of a Simple Generation

HTML document:

```
 <HTML><HEAD><TITLE>
 Example1 genNat
</TITLE></HEAD><BODY><H2>
 Example1 genNat
 </H2><HR>
 <P>This is for your output
 </BODY></HTML>
```

Generated Natural subprogram:

```
* ----- SUBPROGRAM generated out of file:
* ----- C:\example1.html
DEFINE DATA
PARAMETER USING W3PARM
LOCAL USING W3CONST
LOCAL
* ----- PRIVATE VARIABLES -----
1 W3VALUE (A250)
END-DEFINE
*
* ----- ERROR HANDLER -----
ON ERROR
  PERFORM W3ERROR ##W3ERROR
  PERFORM W3END ##RPC
  ESCAPE ROUTINE
END-ERROR
* ----- INITIALIZE HTTP API -----
PERFORM W3INIT ##RPC
* ----- HEADER FOR SERVER -----
PERFORM W3CONTENT-TYPE 'text/html'
*
* ----- MAIN PROGRAM -----
PERFORM W3TEXTLINE'<HTML><HEAD><TITLE>'
PERFORM W3TEXTLINE 'Example genNat'
PERFORM W3TEXTLINE'</TITLE></HEAD><BODY><H2>'
PERFORM W3TEXTLINE 'Example genNat'
PERFORM W3TEXTLINE '</H2><HR>'
PERFORM W3TEXTLINE '<P>This is for your output'
PERFORM W3TEXTLINE '</BODY></HTML> '
* ----- END HTTP API -----
PERFORM W3END ##RPC
* ----- END MAIN PROGRAM -----
*
*
* ----- SUBROUTINES -----
END
```

Example 2 of a Simple Generation with a Natural Tag

HTML document:

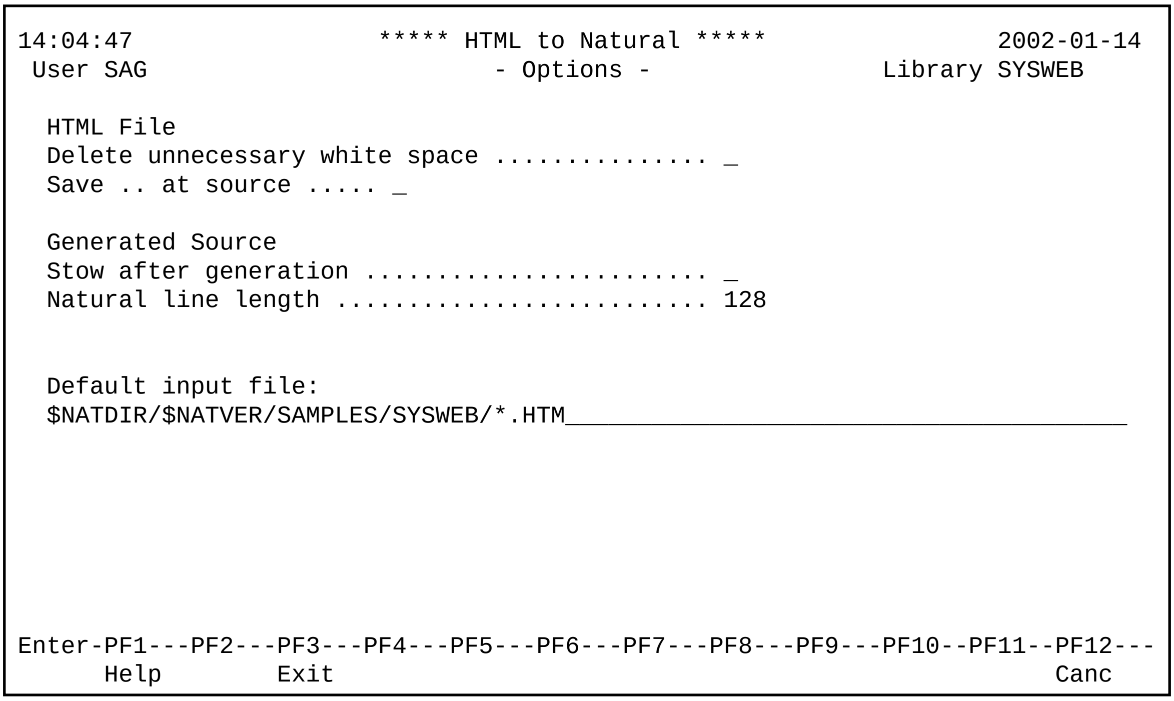
```
  <HTML><HEAD><TITLE>
  Example2 genNat
 </TITLE></HEAD><BODY><H2>
  Example2 genNat
  </H2><HR>
  <P>This is for your output
  <HR>
  <P>generated at:
  <NATURAL NOT>
  Time/Date
  </NATURAL>
  <NATURAL><!--
   PERFORM DOTIME
  --></NATURAL>
  <NATURAL SUB><!--
  DEFINE SUBROUTINE DOTIME
   COMPRESS *TIME *DATE INTO #VALUE
   PERFORM W3TEXTLINE #VALUE
  END-SUBROUTINE
  --></NATURAL>
  <NATURAL DATA><!--
  1 #VALUE (A30)
  --></NATURAL>
  </BODY></HTML>
```

Generated Natural subprogram:

```
* ----- SUBPROGRAM generated out of file:
* ----- C:\example2.html
DEFINE DATA
PARAMETER USING W3PARM
LOCAL USING W3CONST
1 #VALUE (A30)
* ----- PRIVATE VARIABLES -----
1 W3VALUE (A250)
END-DEFINE
*
* ----- ERROR HANDLER -----
ON ERROR
  PERFORM W3ERROR ##W3ERROR
  PERFORM W3END ##RPC
  ESCAPE ROUTINE
END-ERROR
* ----- INITIALIZE HTTP API -----
PERFORM W3INIT ##RPC
* ----- HEADER FOR SERVER -----
PERFORM W3CONTENT-TYPE ’text/html’
*
* ----- MAIN PROGRAM -----
PERFORM W3TEXTLINE’<HTML><HEAD><TITLE>’
PERFORM W3TEXTLINE ’Example2 genNat’
PERFORM W3TEXTLINE’</TITLE></HEAD><BODY><H2>’
PERFORM W3TEXTLINE ’Example2 genNat’
PERFORM W3TEXTLINE ’</H2><HR>’
PERFORM W3TEXTLINE ’<P>This is for your output’
PERFORM W3TEXTLINE ’<HR>’
PERFORM W3TEXTLINE ’<P>generated at:’
 PERFORM DOTIME
PERFORM W3TEXTLINE ’</BODY></HTML>’
* ----- END HTTP API -----
PERFORM W3END ##RPC
* ----- END MAIN PROGRAM -----
*
*
* ----- SUBROUTINES -----
DEFINE SUBROUTINE DOTIME
 COMPRESS *TIME *DATE INTO #VALUE
 PERFORM W3TEXTLINE #VALUE
END-SUBROUTINE
END
```

Note:
The syntax of the Natural program will not be checked during conversion.

Options



```
14:04:47                  ***** HTML to Natural *****               2002-01-14
 User SAG                         - Options -               Library SYSWEB

  HTML File
  Delete unnecessary white space ............... _
  Save .. at source ..... _

  Generated Source
  Stow after generation ........................ _
  Natural line length .......................... 128


  Default input file:
  $NATDIR/$NATVER/SAMPLES/SYSWEB/*.HTM_______________________________________







Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit                                                  Canc
```

Below is information on:

- Input/Output Fields
- Functions and PF Keys

Input/Output Fields

Field	Explanation
Delete unnecessary white space	If checked, multiple white-space characters such as blank, new line, tab, will be reduced to a single white space. For special HTML tags such as <PRE> <TEXTAREA> or <SCRIPT>, the white space will not be collapsed. Default value: unchecked
Save <NATURAL NOT> ... <NATURAL> in Source	If checked, the content of <NATURAL NOT> tags will not usually be generated into the Natural source. This option generates the content of <NATURAL NOT> as comment into the Natural source. Default value: unchecked
Stow after Generation	If checked, the generated program will be stowed if the generation has been successful. Default value: checked
Natural Line Length	The length of the generated Natural source lines: the minimum value is 20, the maximum 248. Default value: 72
Default Input File	The default input file to be used for the generation. Default value: /nat/natc/511/samples/sysexxt

Functions and PF Keys

PF Key	Function	Explanation
ENTER		Leaves the program and saves the changes.
PF3	Exit	Returns to the command line.
PF12	Canc	Leaves the program without saving your changes.

Generating a DCOM Class

If the Natural Web Interface subprograms should be called using DCOM instead of RPC, a DCOM class is needed. This class contains as methods all relevant Natural subprograms for the Natural Web Interface.

The program HTML to Natural automatically generates the specified class. To stow the generated class, a Local Data Area (LDA) is needed to specify the Global Unique IDs (GUIDs) of the DCOM objects. The name of the LDA begins with **L** which is followed by the first seven characters of Library Name.

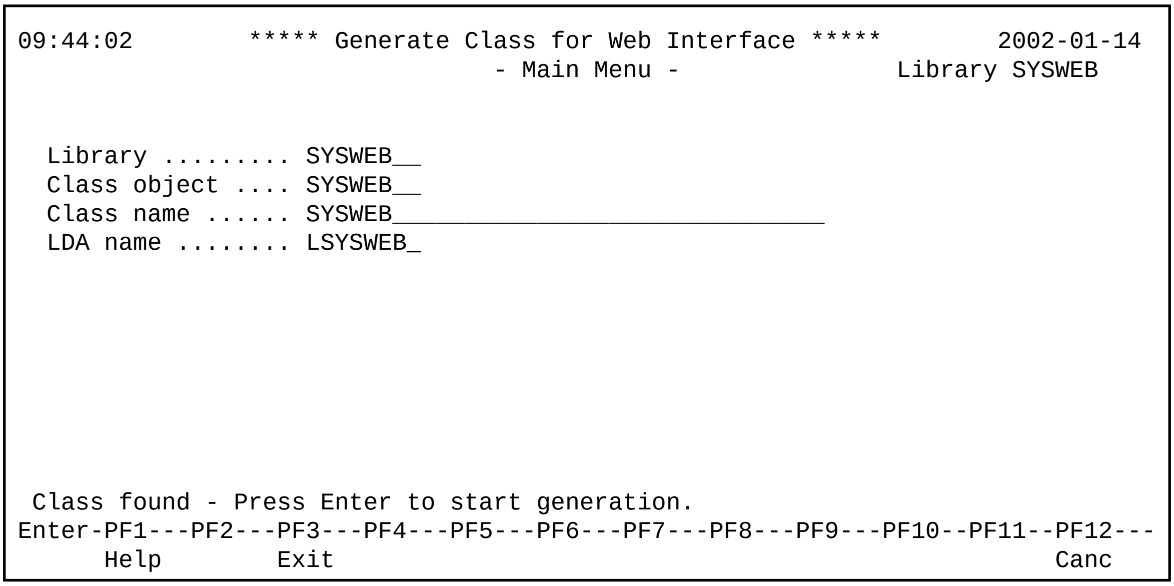
Below is information on:

- Invoking Generate Class
- Input/Output Fields
- Example for Library SYSWEB
- Functions and PF Keys

Invoking Generate Class

To invoke the screen Generate Class for Web Interface

1. Start program WEB-R2DC
   Or open program HTML2NAT.
2. Press PF6.



```
09:44:02        ***** Generate Class for Web Interface *****        2002-01-14
                                - Main Menu -              Library SYSWEB


  Library ......... SYSWEB__
  Class object .... SYSWEB__
  Class name ...... SYSWEB______________________________
  LDA name ........ LSYSWEB_










 Class found - Press Enter to start generation.
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit                                                     Canc
```

Input/Output Fields

Field	Explanation
Library	The name of the library to be scanned.
Class Object	The name of the class source. We recommend that the name you choose for Class Source is identical to the name of the library.
Class Name	The name of the class that can be called later from the Internet. We recommend that the name you choose for Class Name is identical to the name of the library for which the class is generated.
LDA Name	The name of the LDA containing the GUIDs for the class ID and the Natural Web Interface ID. For the naming conventions that apply, see Example for Library SYSWEB below.

Example for Library SYSWEB

The LDA name is LSYSWEB. Name the first GUID CLSID- followed by the Library name and the second GUID IID-NATWEB.

	T		Comment			
			*** Top of Data Area ***			
X	U	1	CLSID-**SYSWEB**	A	36	
X	U	1	IID-NATWEB	A	36	
			*** End of Data Area ***			

Attention:
Do **not** copy and rename or move an LDA in order to get new GUIDs for your classes. If an LDA is copied and renamed or moved, the preset GUID is not changed. This may cause major problems.

Functions and PF Keys

PF Key	Function	Explanation
ENTER		Generates the class and leaves the program.
PF3	Exit	Returns to the command line.
PF4	Defau	Sets default values. This function is enabled if no relevant class is found for the library. The defaults for class source and class name are given. The LDA needed has to be generated in advance.
PF12	Canc	Leaves the program without generation.

Online Test Utility WEB-ONL

This Test Utility is a component of Natural Web Interface. You have the ability to check your subprogram locally without involving an HTTP server. The transfer parameters for your Web page are transferred into the Test Utility and are posted directly to the business logic. As communication platform, you can choose either RPC or DCOM as in real remote communications. The result is either the Web page expected or an error message. The Web page can be viewed with the browser or a viewer of your choice. If you receive an error message, you can easily debug your business logic locally without writing an extra test routine. No remote debugging is needed.

Features:

- Local application check.
- No need for remote debugging.
- Simplified error checking.
- No need to write an extra test routine.

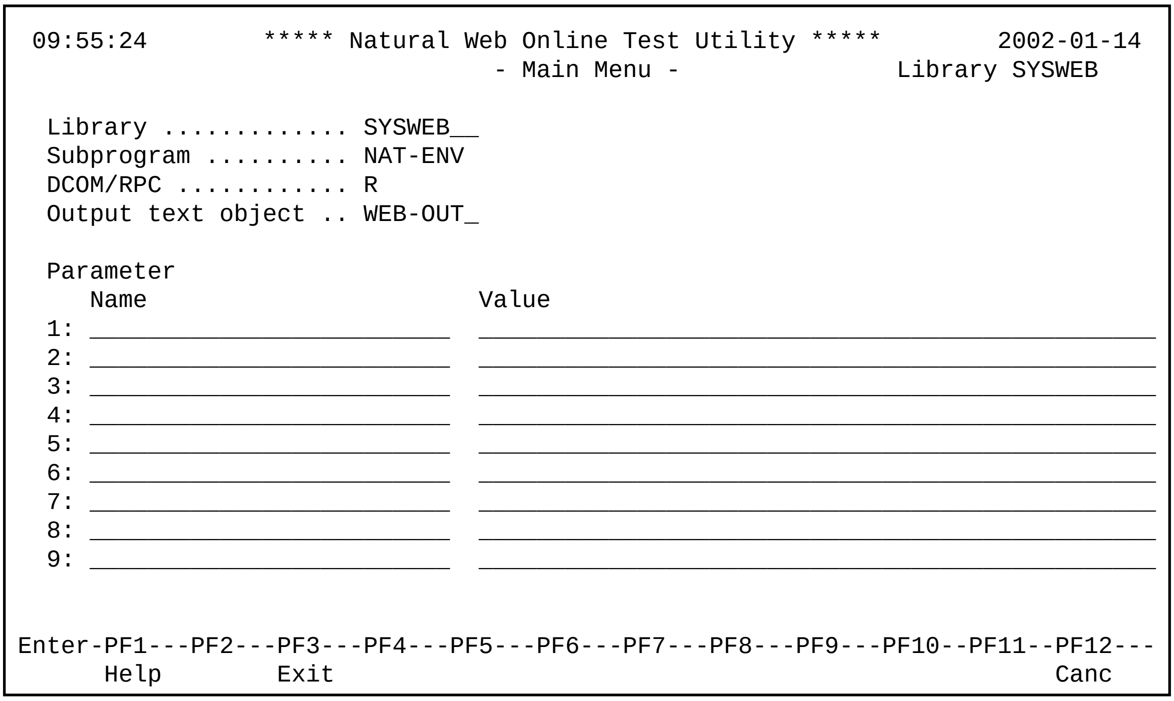
Below is information on:

- Running the Application
- Input/Output Fields
- Functions and PF Keys

Running the Application

To run the application

1. Start the program WEB-ONL.
2. Select a library and subprogram name.
3. Optional: add parameters.
4. Choose RPC or DCOM.
5. Press ENTER.



```
 09:55:24        ***** Natural Web Online Test Utility *****        2002-01-14
                                 - Main Menu -               Library SYSWEB

  Library ............. SYSWEB__
  Subprogram .......... NAT-ENV
  DCOM/RPC ............ R
  Output text object .. WEB-OUT_

  Parameter
     Name                           Value
  1: _________________________  _______________________________________________
  2: _________________________  _______________________________________________
  3: _________________________  _______________________________________________
  4: _________________________  _______________________________________________
  5: _________________________  _______________________________________________
  6: _________________________  _______________________________________________
  7: _________________________  _______________________________________________
  8: _________________________  _______________________________________________
  9: _________________________  _______________________________________________


Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit                                                  Canc
```

Input/Output Fields

Field	Explanation
Library	The library in which the required subprogram is stored.
Subprogram	The name of the required subprogram.
DCOM/RPC	Can be selected with either DCOM or RPC as communication form. For DCOM, you have to register your classes first. Default: R
Output Text Object	The name of the Natural object of the type Text that stores the result of the generated Web page. Default: WEB-OUT
Parameter: Name Value	Here you can enter the name-value-pairs needed from the subprogram. If you use server parameters, place an ampersand (&) in front of the variable name before you add the parameter to the parameter list.

Functions and PF Keys

PF Key	Function	Explanation
ENTER		Runs the process of receiving the output of the requested subprogram. The status of the process can be seen in the message line at the bottom of the WEB-ONL program screen.
PF3	Exit	Leaves the Test Utility and returns to the command line.
PF12	Canc	Cancel. Stops processing.